```
                        title   'Root module of relocatable BIOS for CP/M 3.0'

                        ; version 1.0 15 Sept 82

FFFF =          true    equ -1
0000 =          false   equ not true

FFFF =          banked  equ true


                ;               Copyright (C), 1982
                ;               Digital Research, Inc
                ;                  P.O. Box 579
                ;              Pacific Grove, CA  93950


                ;   This is the invariant portion of the modular BIOS and is
                ;       distributed as source for informational purposes only.
                ;       All desired modifications should be performed by
                ;       adding or changing externally defined modules.
                ;       This allows producing "standard" I/O modules that
                ;       can be combined to support a particular system
                ;       configuration.

000D =          cr      equ 13
000A =          lf      equ 10
0007 =          bell    equ 7
0011 =          ctlQ    equ 'Q'-'@'
0013 =          ctlS    equ 'S'-'@'

0100 =          ccp     equ 0100h       ; Console Command Processor gets loaded into the TPA

                        cseg            ; GENCPM puts CSEG stuff in common memory


                    ; variables in system data page

                        extrn @covec,@civec,@aovec,@aivec,@lovec ; I/O redirection vectors
                        extrn @mxtpa                            ; addr of system entry point
                        extrn @bnkbf                            ; 128 byte scratch buffer

                    ; initialization

                        extrn ?init                     ; general initialization and signon
                        extrn ?ldccp,?rlccp             ; load & reload CCP for BOOT & WBOOT

                    ; user defined character I/O routines

                        extrn ?ci,?co,?cist,?cost       ; each take device in <B>
                        extrn ?cinit                    ; (re)initialize device in <C>
                        extrn @ctbl                     ; physical character device table

                    ; disk communication data items

                        extrn @dtbl                     ; table of pointers to XDPHs
                        public @adrv,@rdrv,@trk,@sect   ; parameters for disk I/O
```

CP/M-PLUS V.3.0
COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

```
                        public @dma,@dbnk,@cnt          ;   ''      ''  ''  '.
                    ; memory control
                        public @cbnk                    ; current bank
                        extrn ?xmove,?move              ; select move bank, and block move
                        extrn ?bank                     ; select CPU bank
                    ; clock support
                        extrn ?time                     ; signal time operation
                    ; general utility routines
                        public ?pmsg,?pdec      ; print message, print number from 0 to 65535
                        public ?pderr           ; print BIOS disk error message header
                        maclib modebaud         ; define mode bits
                    ; External names for BIOS entry points
                        public ?boot,?wboot,?const,?conin,?cono,?list,?auxo,?auxi
                        public ?home,?sldsk,?sttrk,?stsec,?stdma,?read,?write
                        public ?lists,?sctrn
                        public ?conos,?auxis,?auxos,?dvtbl,?devin,?drtbl
                        public ?mltio,?flush,?mov,?tim,?bnksl,?stbnk,?xmov
                    ; BIOS Jump vector.
                            ; All BIOS routines are invoked by calling these
                            ;       entry points.
0000 C30000     ?boot:  jmp boot        ; initial entry on cold start
0003 C36C00     ?wboot: jmp wboot       ; reentry on program exit, warm start
0006 C37701     ?const: jmp const       ; return console input status
0009 C39201     ?conin: jmp conin       ; return console input character
000C C3DA00     ?cono:  jmp conout      ; send console output character
000F C3E600     ?list:  jmp list        ; send list output character
0012 C3E000     ?auxo:  jmp auxout      ; send auxilliary output character
0015 C39801     ?auxi:  jmp auxin       ; return auxilliary input character
001B C36E00     ?home:  jmp home        ; set disks to logical home
001B C33F00     ?sldsk: jmp seldsk      ; select disk drive, return disk parameter info
001E C37100     ?sttrk: jmp settrk      ; set disk track
0021 C37700     ?stsec: jmp setsec      ; set disk sector
0024 C37D00     ?stdma: jmp setdma      ; set disk I/O memory address
0027 C39400     ?read:  jmp read        ; read physical block(s)
002A C3AA00     ?write: jmp write       ; write physical block(s)
002D C31201     ?lists: jmp listst      ; return list device status
0030 C38900     ?sctrn: jmp sectrn      ; translate logical to physical sector
0033 C30601     ?conos: jmp conost      ; return console output status
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #:

```
                ?auxis: jmp auxist      ; return aux input status
                ?auxos: jmp auxost      ; return aux output status
                ?dvtbl: jmp devtbl      ; return address of device def table
                ?devin: jmp ?cinit      ; change baud rate of device

                ?drtbl: jmp getdrv      ; return address of disk drive table
                ?mltio: jmp multio      ; set multiple record count for disk I/O
                ?flush: jmp flush       ; flush BIOS maintained disk caching

                ?mov:   jmp ?move       ; block move memory to memory
                ?tim:   jmp ?time       ; Signal Time and Date operation
                ?bnksl: jmp bnksel      ; select bank for code execution and default DMA
                ?stbnk: jmp setbnk      ; select different bank for disk I/O DMA operations.
                ?xmov:  jmp ?xmove      ; set source and destination banks for one operation

                        jmp 0           ; reserved for future expansion
                        jmp 0           ; reserved for future expansion
                        jmp 0           ; reserved for future expansion


                        ; BOOT
                        ;       Initial entry point for system startup.

                        dseg    ; this part can be banked

                boot:
0000 31D200             lxi sp,boot$stack
0003 0E0F               mvi c,15        ; initialize all 16 character devices
                c$init$loop:
0005 C5CD0000C1         push b ! call ?cinit ! pop b
000A 0DF20500           dcr c ! jp c$init$loop

000E CD0000             call ?init      ; perform any additional system initialization
                                        ; and print signon message

0011 0100102100         lxi b,16*256+0 ! lxi h,@dtbl    ; init all 16 logical disk drives
                d$init$loop:
0017 C5                 push b          ; save remaining count and abs drive
0018 5E235623           mov e,m ! inx h ! mov d,m ! inx h       ; grab @drv entry
001C 7BB2CA3600         mov a,e ! ora d ! jz d$init$next        ; if null, no drive
0021 E5                 push h                                  ; save @drv pointer
0022 EB                 xchg                                    ; XDPH address in <HL>
0023 2B2B7E32EE         dcx h ! dcx h ! mov a,m ! sta @RDRV     ; get relative drive code
0029 7932ED00           mov a,c ! sta @ADRV                     ; get absolute drive code
002D 2B                 dcx h                                   ; point to init pointer
002E 562B5E             mov d,m ! dcx h ! mov e,m               ; get init pointer
0031 EBCDB601           xchg ! call ipchl                       ; call init routine
0035 E1                 pop h                                   ; recover @drv pointer
                d$init$next:
0036 C1                 pop b                                   ; recover counter and drive #
0037 0C05C21700         inr c ! dcr b ! jnz d$init$loop          ; and loop for each drive
003C C36300             jmp boot$1

                        cseg    ; following in resident memory

                boot$1:
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                        call set$jumps
0063 CD7800             call ?ldccp                     ; fetch CCP for first time
0066 CD0000             jmp ccp
0069 C30001
                        ; WBOOT
                        ;       Entry for system restarts.
                wboot:
006C 31D200             lxi sp,boot$stack
006F CD7800             call set$jumps          ; initialize page zero
0072 CD0000             call ?rlccp             ; reload CCP
0075 C30001             jmp ccp                 ; then reset jmp vectors and exit to ccp
                set$jumps:
                  if banked
0078 3E01CD5100         mvi a,1 ! call ?bnksl
                  endif
007D 3EC3               mvi a,JMP
007F 3200003205         sta 0 ! sta 5           ; set up jumps in page zero
0085 2103002201         lxi h,?wboot ! shld 1   ; BIOS warm start entry
008B 2A00002206         lhld @MXTPA ! shld 6    ; BDOS system call entry
0091 C9                 ret
0092                            ds 64
00D2 =          boot$stack      equ $
                        ; DEVTBL
                        ;       Return address of character device table
                devtbl:
00D2 210000C9           lxi h,@ctbl ! ret
                        ; GETDRV
                        ;       Return address of drive table
                getdrv:
00D6 210000C9           lxi h,@dtbl ! ret
                        ; CONOUT
                        ;       Console Output.  Send character in <C>
                        ;                       to all selected devices
                conout:
00DA 2A0000             lhld @covec     ; fetch console output bit vector
00DD C3E900             jmp out$scan
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        ; AUXOUT
                        ;       Auxiliary Output. Send character in <C>
                        ;                       to all selected devices
                
                auxout:
00E0 2A0000             lhld @aovec     ; fetch aux output bit vector
00E3 C3E900             jmp out$scan
                
                        ; LIST
                        ;       List Output.  Send character in <C>
                        ;                       to all selected devices.
                
                list:
00E6 2A0000             lhld @lovec     ; fetch list output bit vector
                
                out$scan:
00E9 0600               mvi b,0         ; start with device 0
                co$next:
00EB 29                 dad h           ; shift out next bit
00EC D2FF00             jnc not$out$device
00EF E5                 push h          ; save the vector
00F0 C5                 push b          ; save the count and character
                not$out$ready:
00F1 CD2C01B7CA         call coster ! ora a ! jz not$out$ready
00F8 C1C5               pop b ! push b  ; restore and resave the character and device
00FA CD0000             call ?co        ; if device selected, print it
00FD C1                 pop b           ; recover count and character
00FE E1                 pop h           ; recover the rest of the vector
                not$out$device:
00FF 04                 inr b           ; next device number
0100 7CB5               mov a,h ! ora l ; see if any devices left
0102 C2EB00             jnz co$next     ; and go find them...
0105 C9                 ret
                
                        ; CONOST
                        ;       Console Output Status.  Return true if
                        ;               all selected console output devices
                        ;               are ready.
                
                conost:
0106 2A0000             lhld @covec     ; get console output bit vector
0109 C31501             jmp ost$scan
                
                        ; AUXOST
                        ;       Auxiliary Output Status.  Return true if
                        ;               all selected auxiliary output devices
                        ;               are ready.
                
                auxost:
010C 2A0000             lhld @aovec     ; get aux output bit vector
010F C31501             jmp ost$scan
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                        ; LISTST
                        ;       List Output Status.  Return true if
                        ;               all selected list output devices
                        ;               are ready.
                
                listst:
0112 2A0000             lhld @lovec     ; get list output bit vector
                
                ost$scan:
0115 0600               mvi b,0         ; start with device 0
                cos$next:
0117 29                 dad h           ; check next bit
0118 E5                 push h          ; save the vector
0119 C5                 push b          ; save the count
011A 3EFF               mvi a,0FFh      ; assume device ready
011C DC2C01             cc coster       ; check status for this device
011F C1                 pop b           ; recover count
0120 E1                 pop h           ; recover bit vector
0121 B7                 ora a           ; see if device ready
0122 C8                 rz              ; if any not ready, return false
0123 04                 inr b           ; drop device number
0124 7CB5               mov a,h ! ora l ; see if any more selected devices
0126 C21701             jnz cos$next
0129 F6FF               ori 0FFh        ; all selected were ready, return true
012B C9                 ret
                
                coster:         ; check for output device ready, including optional
                                ;       xon/xoff support
012C 682600             mov l,b ! mvi h,0       ; make device code 16 bits
012F E5                 push h                  ; save it in stack
0130 292929             dad h ! dad h ! dad h   ; create offset into device characteristics tbl
0133 11060019           lxi d,@ctbl+6 ! dad d   ; make address of mode byte
0137 7EE610             mov a,m ! ani mb$xonxoff
013A E1                 pop h                   ; recover console number in <HL>
013B CA0000             jz ?cost                ; not a xon device, go get output status direct
013E 112B0219           lxi d,xofflist ! dad d  ; make pointer to proper xon/xoff flag
0142 CD5D01             call cist1              ; see if this keyboard has character
0145 7EC46F01           mov a,m ! cnz ci1       ; get flag or read key if any
0149 FE11C25001         cpi ctlq ! jnz not$q    ; if its a ctl-Q,
014E 3EFF               mvi a,0FFh              ;       set the flag ready
                not$q:
0150 FE13C25701         cpi ctls ! jnz not$s    ; if its a ctl-S,
0155 3E00               mvi a,00h               ;       clear the flag
                not$s:
0157 77                 mov m,a                 ; save the flag
0158 CD6601             call cost1              ; get the actual output status,
015B A6                 ana m                   ; and mask with ctl-Q/ctl-S flag
015C C9                 ret                     ; return this as the status
                
                cist1:                  ; get input status with <BC> and <HL> saved
015D C5E5               push b ! push h
015F CD0000             call ?cist
0162 E1C1               pop h ! pop b
0164 B7                 ora a
0165 C9                 ret
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                        
                cost1:                  ; get output status, saving <BC> & <HL>
0166 C5E5               push b ! push h
016B CD0000             call ?cost
016B E1C1               pop h ! pop b
016D B7                 ora a
016E C9                 ret
                        
                ci1:                    ; get input, saving <BC> & <HL>
016F C5E5               push b ! push h
0171 CD0000             call ?ci
0174 E1C1               pop h ! pop b
0176 C9                 ret
                        
                        
                        ; CONST
                        ;       Console Input Status.  Return true if
                        ;               any selected console input device
                        ;               has an available character.
                        
                const:
0177 2A0000             lhld @civec     ; get console input bit vector
017A C38001             jmp ist$scan
                        
                        
                        ; AUXIST
                        ;       Auxiliary Input Status.  Return true if
                        ;               any selected auxiliary input device
                        ;               has an available character.
                        
                auxist:
017D 2A0000             lhld @aivec     ; get aux input bit vector
                        
                ist$scan:
0180 0600               mvi b,0         ; start with device 0
                cis$next:
0182 29                 dad h           ; check next bit
0183 3E00               mvi a,0         ; assume device not ready
0185 DC5D01             cc cist1        ; check status for this device
0188 B7C0               ora a ! rnz     ; if any ready, return true
018A 04                 inr b           ; drop device number
018B 7CB5               mov a,h ! ora l ; see if any more selected devices
018D C28201             jnz cis$next
0190 AF                 xra a           ; all selected were not ready, return false
0191 C9                 ret
                        
                        
                        ; CONIN
                        ;       Console Input.  Return character from first
                        ;               ready console input device.
                        
                conin:
0192 2A0000             lhld @civec
0195 C39B01             jmp in$scan
                        
                        
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
                        ; AUXIN
                        ;       Auxiliary Input.  Return character from first
                        ;               ready auxiliary input device.
                
                auxin:
0198 2A0000             lhld @aivec
                
                in$scan:
019B E5                 push h          ; save bit vector
019C 0600               mvi b,0
                ci$next:
019E 29                 dad h           ; shift out next bit
019F 3E00               mvi a,0         ; insure zero a  (nonexistant device not ready).
01A1 DC5D01             cc cist1        ; see if the device has a character
01A4 B7                 ora a
01A5 C2B201             jnz ci$rdy      ; this device has a character
01A8 04                 inr b           ; else, next device
01A9 7CB5               mov a,h ! ora l ; see if any more devices
01AB C29E01             jnz ci$next     ; go look at them
01AE E1                 pop h           ; recover bit vector
01AF C39B01             jmp in$scan     ; loop til we find a character
                
                ci$rdy:
01B2 E1                 pop h           ; discard extra stack
01B3 C30000             jmp ?ci
                
                
                ;       Utility Subroutines
                
                
                ipchl:          ; vectored CALL point
01B6 E9                 pchl
                
                
                ?pmsg:          ; print message @<HL> up to a null
                                ; saves <BC> & <DE>
01B7 C5                 push b
01B8 D5                 push d
                pmsg$loop:
01B9 7EB7CAC801         mov a,m ! ora a ! jz pmsg$exit
01BE 4FE5               mov c,a ! push h
01C0 CD0C00E1           call ?cono ! pop h
01C4 23C3B901           inx h ! jmp pmsg$loop
                pmsg$exit:
01C8 D1                 pop d
01C9 C1                 pop b
01CA C9                 ret
                
                ?pdec:          ; print binary number 0-65535 from <HL>
01CB 01F30111F0         lxi b,table10! lxi d,-10000
                next:
01D1 3E2F               mvi a,'0'-1
                pdecl:
01D3 E53C19D2DE         push h! inr a! dad d! jnc stoploop
01D9 3333C3D301         inx sp! inx sp! jmp pdecl
                stoploop:
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 449  01DE D5C5              push d! push b
 450  01E0 4FCD0C00          mov c,a! call ?cono
 451  01E4 C1D1              pop b! pop d
 452                 nextdigit:
 453  01E6 E1                pop h
 454  01E7 0A5F03            ldax b! mov e,a! inx b
 455  01EA 0A5703            ldax b! mov d,a! inx b
 456  01ED 7BB2C2D101        mov a,e! ora d! jnz next
 457  01F2 C9                ret
 458
 459                 table10:
 460  01F3 18FC9CFFF6        dw      -1000,-100,-10,-1,0
 461
 462                 ?pderr:
 463  01FD 21D100CDB7        lxi h,drive$msg ! call ?pmsg                    ; error header
 464  0203 3AED00C641        lda @adrv ! adi 'A' ! mov c,a ! call ?cono       ; drive code
 465  020C 21E300CDB7        lxi h,track$msg ! call ?pmsg                    ; track header
 466  0212 2AEF00CDCB        lhld @trk ! call ?pdec                          ; track number
 467  0218 21E800CDB7        lxi h,sector$msg ! call ?pmsg                   ; sector header
 468  021E 2AF100CDCB        lhld @sect ! call ?pdec                         ; sector number
 469  0224 C9                ret
 470
 471
 472                         ; BNKSEL
 473                         ;       Bank Select.  Select CPU bank for further execution.
 474
 475                 bnksel:
 476  0225 323B02            sta @cbnk                       ; remember current bank
 477  0228 C30000            jmp ?bank                       ; and go exit through users
 478                                                         ; physical bank select routine
 479
 480
 481  022B FFFFFFFFFFxofflist        db      -1,-1,-1,-1,-1,-1,-1,-1         ; ctl-s clears to zero
 482  0233 FFFFFFFFFF                db      -1,-1,-1,-1,-1,-1,-1,-1
 483
 484
 485
 486                         dseg    ; following resides in banked memory
 487
 488
 489
 490                 ;       Disk I/O interface routines
 491
 492
 493                         ; SELDSK
 494                         ;       Select Disk Drive.  Drive code in <C>.
 495                         ;               Invoke login procedure for drive
 496                         ;               if this is first select.  Return
 497                         ;               address of disk parameter header
 498                         ;               in <HL>
 499
 500                 seldsk:
 501  003F 7932ED00          mov a,c ! sta @adrv                     ; save drive select code
 502  0043 69260029          mov l,c ! mvi h,0 ! dad h               ; create index from drive code
 503  0047 01000009          lxi b,@dtbl ! dad b                     ; get pointer to dispatch table
 504  004B 7E23666F          mov a,m ! inx h ! mov h,m ! mov l,a      ; point at disk descriptor
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                ora h ! rz                              ; if no entry in table, no disk
0051 7BE601C26D                 mov a,e ! ani 1 ! jnz not$first$select  ; examine login bit
0057 E5EB                       push h ! xchg                           ; put pointer in stack & <DE>
0059 21FEFF197E                 lxi h,-2 ! dad d ! mov a,m ! sta @RDRV  ; get relative drive
0061 21FAFF19                   lxi h,-6 ! dad d                        ; find LOGIN addr
0065 7E23666F                   mov a,m ! inx h ! mov h,m ! mov l,a     ; get address of LOGIN routine
0069 CDB601                     call ipchl                              ; call LOGIN
006C E1                         pop h                                   ; recover DPH pointer
                        not$first$select:
006D C9                         ret


                                ; HOME
                                ;       Home selected drive.  Treated as SETTRK(0).

                        home:
006E 010000                     lxi b,0         ; same as set track zero


                                ; SETTRK
                                ;       Set Track. Saves track address from <BC>
                                ;               in @TRK for further operations.

                        settrk:
0071 6960                       mov l,c ! mov h,b
0073 22EF00                     shld @trk
0076 C9                         ret


                                ; SETSEC
                                ;       Set Sector.  Saves sector number from <BC>
                                ;               in @sect for further operations.

                        setsec:
0077 6960                       mov l,c ! mov h,b
0079 22F100                     shld @sect
007C C9                         ret


                                ; SETDMA
                                ;       Set Disk Memory Address.  Saves DMA address
                                ;               from <BC> in @DMA and sets @DBNK to @CBNK
                                ;               so that further disk operations take place
                                ;               in current bank.

                        setdma:
007D 6960                       mov l,c ! mov h,b
007F 22F300                     shld @dma

0082 3A3B02                     lda @cbnk       ; default DMA bank is current bank
                                                ; fall through to set DMA bank


                                ; SETBNK
                                ;       Set Disk Memory Bank.  Saves bank number
                                ;               in @DBNK for future disk data
                                ;               transfers.
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                        setbnk:
0085 32F600                     sta @dbnk
0088 C9                         ret

                        ; SECTRN
                        ;       Sector Translate.  Indexes skew table in <DE>
                        ;               with sector in <BC>.  Returns physical sector
                        ;               in <HL>.  If no skew table (<DE>=0) then
                        ;               returns physical=logical.

                        sectrn:
0089 6960                       mov l,c ! mov h,b
008B 7AB3C8                     mov a,d ! ora e ! rz
008E EB096E2600                 xchg ! dad b ! mov l,m ! mvi h,0
0093 C9                         ret

                        ; READ
                        ;       Read physical record from currently selected drive.
                        ;               Finds address of proper read routine from
                        ;               extended disk parameter header (XDPH).

                        read:
0094 2AED002600                 lhld @adrv ! mvi h,0 ! dad h    ; get drive code and double it
009A 11000019                   lxi d,@dtbl ! dad d             ; make address of table entry
009E 7E23666F                   mov a,m ! inx h ! mov h,m ! mov l,a     ; fetch table entry
00A2 E5                         push h                          ; save address of table
00A3 11F8FF19                   lxi d,-8 ! dad d                ; point to read routine address
00A7 C3BD00                     jmp rw$common                   ; use common code

                        ; WRITE
                        ;       Write physical sector from currently selected drive.
                        ;               Finds address of proper write routine from
                        ;               extended disk parameter header (XDPH).

                        write:
00AA 2AED002600                 lhld @adrv ! mvi h,0 ! dad h    ; get drive code and double it
00B0 11000019                   lxi d,@dtbl ! dad d             ; make address of table entry
00B4 7E23666F                   mov a,m ! inx h ! mov h,m ! mov l,a     ; fetch table entry
00B8 E5                         push h                          ; save address of table
00B9 11F6FF19                   lxi d,-10 ! dad d               ; point to write routine address

                        rw$common:
00BD 7E23666F                   mov a,m ! inx h ! mov h,m ! mov l,a     ; get address of routine
00C1 D1                         pop d                           ; recover address of table
00C2 1B1B                       dcx d ! dcx d                   ; point to relative drive
00C4 1A32EE00                   ldax d ! sta @rdrv              ; get relative drive code and post it
00C8 1313                       inx d ! inx d                   ; point to DPH again
00CA E9                         pchl                            ; leap to driver


                        ; MULTIO
                        ;       Set multiple sector count. Saves passed count in
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER #

```
                        ;               @CNT
                
                multio:
00CB 32F500C9           sta @cnt ! ret
                
                
                        ; FLUSH
                        ;       BIOS deblocking buffer flush.  Not implemented.
                
                flush:
00CF AFC9               xra a ! ret             ; return with no error
                
                
                
                        ; error message components
00D1 0D0A074249 drive$msg       db      cr,lf,bell,'BIOS Error on ',0
00E3 3A20542D00 track$msg       db      ': T-',0
00E8 2C20532D00 sector$msg      db      ', S-',0
                
                
                    ; disk communication data items
                
00ED            @adrv   ds      1               ; currently selected disk drive
00EE            @rdrv   ds      1               ; controller relative disk drive
00EF            @trk    ds      2               ; current track number
00F1            @sect   ds      2               ; current sector number
00F3            @dma    ds      2               ; current DMA address
00F5 00         @cnt    db      0               ; record count for multisector transfer
00F6 00         @dbnk   db      0               ; bank for DMA operations
                
                
                        cseg    ; common memory
                
023B 00         @cbnk   db      0               ; bank for processor operations
                
                
023C                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

AUXIN	0198	99	397#		
AUXIST	017D	113	367#		
AUXOST	010C	114	277#		
AUXOUT	00E0	98	230#		
BANKED	FFFF	8#	186		
BAUD110	0003				
BAUD1200	0008				
BAUD134	0004				
BAUD150	0005				
BAUD1800	0009				
BAUD19200	000F				
BAUD2400	000A				
BAUD300	0006				
BAUD3600	000B				
BAUD4800	000C				
BAUD50	0001				
BAUD600	0007				
BAUD7200	000D				
BAUD75	0002				
BAUD9600	000E				
BAUDNONE	0000				
BELL	0007	27#	632		
BNKSEL	0225	124	475#		
BOOT	0000	91	138#		
BOOT1	0063	164	168#		
BOOTSTACK	00D2	139	178	198#	
CCP	0100	31#	171	181	
CI1	016F	319	345#		
CINEXT	019E	403#	411		
CINITLOOP	0005	141#	143		
CIRDY	01B2	408	415#		
CISNEXT	0182	372#	379		
CIST1	015D	318	331#	375	406
CONEXT	00EB	244#	258		
CONIN	0192	95	388#		
CONOST	0106	112	267#		
CONOUT	00DA	96	220#		
CONST	0177	94	357#		
COSNEXT	0117	292#	304		
COST1	0166	327	338#		
COSTER	012C	250	297	308#	
CR	000D	25#	632		
CTLQ	0011	28#	320		
CTLS	0013	29#	323		
DEVTBL	00D2	115	204#		
DINITLOOP	0017	149#	163		
DINITNEXT	0036	152	161#		
DRIVEMSG	00D1	463	632#		
FALSE	0000	6#			
FLUSH	00CF	120	626#		
GETDRV	00D6	118	211#		
HOME	006E	101	520#		
INSCAN	019B	390	400#	413	
IPCHL	01B6	159	423#	511	
ISTSCAN	0180	359	370#		
LF	000A	26#	632		

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

LIST	00E6	97	239#			
LISTST	0112	109	287#			
MBINOUT	0003					
MBINPUT	0001					
MBOUTPUT	0002					
MBSERIAL	0008					
MBSOFTBAUD	0004					
MBXONXOFF	0010	314				
MULTIO	00CB	119	619#			
NEXT	01D1	443#	456			
NEXTDIGIT	01E6	452#				
NOTFIRSTSELECT	006D	506	513#			
NOTOUTDEVICE	00FF	246	255#			
NOTOUTREADY	00F1	249#	250			
NOTQ	0150	320	322#			
NOTS	0157	323	325#			
OSTSCAN	0115	269	279	290#		
OUTSCAN	00E9	223	232	242#		
PDECL	01D3	445#	447			
PMSGEXIT	01C8	432	436#			
PMSGLOOP	01B9	431#	435			
READ	0094	106	585#			
RWCOMMON	00BD	591	606#			
SECTORMSG	00E8	467	634#			
SECTRN	0089	110	573#			
SELDSK	003F	102	500#			
SETBNK	0085	125	562#			
SETDMA	007D	105	550#			
SETJUMPS	0078	169	179	184#		
SETSEC	0077	104	538#			
SETTRK	0071	103	528#			
STOPLOOP	01DE	446	448#			
TABLE10	01F3	442	459#			
TRACKMSG	00E3	465	633#			
TRUE	FFFF	5#	6	8		
WBOOT	006C	92	177#			
WRITE	00AA	107	599#			
XOFFLIST	022B	317	481#			
?AUXI	0015	79	99#			
?AUXIS	0036	82	113#			
?AUXO	0012	79	98#			
?AUXOS	0039	82	114#			
?BANK	0000	63	477			
?BNKSL	0051	83	124#	187		
?BOOT	0000	79	91#			
?CI	0000	49	347	417		
?CINIT	0000	50	116	142		
?CIST	0000	49	333			
?CO	0000	49	252			
?CONIN	0009	79	95#			
?CONO	000C	79	96#	434	450	464
?CONOS	0033	82	112#			
?CONST	0006	79	94#			
?COST	0000	49	316	340		
?DEVIN	003F	82	116#			
?DRTBL	0042	82	118#			

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

?DVTBL	003C	82	115#					
?FLUSH	0048	83	120#					
?HOME	0018	80	101#					
?INIT	0000	44	145					
?LDCCP	0000	45	170					
?LIST	000F	79	97#					
?LISTS	002D	81	109#					
?MLTIO	0045	83	119#					
?MOV	004B	83	122#					
?MOVE	0000	62	122					
?PDEC	01CB	71	441#	466	468			
?PDERR	01FD	72	462#					
?PMSG	01B7	71	427#	463	465	467		
?READ	0027	80	106#					
?RLCCP	0000	45	180					
?SCTRN	0030	81	110#					
?SLDSK	001B	80	102#					
?STBNK	0054	83	125#					
?STDMA	0024	80	105#					
?STSEC	0021	80	104#					
?STTRK	001E	80	103#					
?TIM	004E	83	123#					
?TIME	0000	67	123					
?WBOOT	0003	79	92#	192				
?WRITE	002A	80	107#					
?XMOV	0057	83	126#					
?XMOVE	0000	62	126					
@ADRV	00ED	56	156	464	501	586	600	639#
@AIVEC	0000	38	368	398				
@AOVEC	0000	38	231	278				
@BNKBF	0000	40						
@CBNK	023B	61	476	554	650#			
@CIVEC	0000	38	358	389				
@CNT	00F5	57	620	644#				
@COVEC	0000	38	222	268				
@CTBL	0000	51	205	313				
@DBNK	00F6	57	563	645#				
@DMA	00F3	57	552	643#				
@DTBL	0000	55	148	212	503	587	601	
@LOVEC	0000	38	240	288				
@MXTPA	0000	39	193					
@RDRV	00EE	56	155	508	610	640#		
@SECT	00F1	56	468	540	642#			
@TRK	00EF	56	466	530	641#			

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                        title   'Boot loader module for CP/M 3.0'

FFFF =          true equ -1
0000 =          false equ not true

FFFF =          banked  equ true

                        public  ?init,?ldccp,?rlccp,?time
                        extrn   ?pmsg,?conin
                        extrn   @civec,@covec,@aivec,@aovec,@lovec
                        extrn   @cbnk,?bnksl

                        maclib ports
                        maclib z80

0005 =          bdos    equ 5

                        if banked
0001 =          tpa$bank        equ 1
                        else
                tpa$bank        equ 0
                        endif

                        dseg    ; init done from banked memory

                ?init:
0000 2100802200         lxi h,08000h ! shld @civec ! shld @covec        ; assign console to CRT:
0009 2100402200         lxi h,04000h ! shld @lovec                      ; assign printer to LPT:
000F 2100202200         lxi h,02000h ! shld @aivec ! shld @aovec        ; assign AUX to CRT1:
0018 21EF00CD25         lxi h,init$table ! call out$blocks      ; set up misc hardware
001E 218700CD00         lxi h,signon$msg ! call ?pmsg           ; print signon message
0024 C9                 ret

                out$blocks:
0025 7EB7C847           mov a,m ! ora a ! rz ! mov b,a
0029 234E23             inx h ! mov c,m ! inx h
                        outir
002C+EDB3               DB      0EDH,0B3H
002E C32500             jmp out$blocks


                        cseg    ; boot loading most be done from resident memory

                    ;   This version of the boot loader loads the CCP from a file
                    ;   called CCP.COM on the system drive (A:).


                ?ldccp:
                        ; First time, load the A:CCP.COM file into TPA
0000 AF32DB00           xra a ! sta ccp$fcb+15          ; zero extent
0004 21000022EC         lxi h,0 ! shld fcb$nr           ; start at beginning of file
000A 11CC00CD73         lxi d,ccp$fcb ! call open       ; open file containing CCP
0010 3CCA4A00           inr a ! jz no$CCP               ; error if no file...
0014 110001CD78         lxi d,0100h ! call setdma       ; start of TPA
001A 118000CD7D         lxi d,128 ! call setmulti       ; allow up to 16K bytes
0020 11CC00CD82         lxi d,ccp$fcb ! call read       ; load the thing
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                                                    ; now,
                                                    ;   copy CCP to bank 0 for reloading
0026 2100010100         lxi h,0100h ! lxi b,0C00h  ; clone 3K, just in case
002C 3A0000F5           lda @cbnk ! push psw       ; save current bank
                ld$1:
0030 3E01CD0000         mvi a,tpa$bank ! call ?bnksl ; select TPA
0035 7EF5               mov a,m ! push psw         ; get a byte
0037 3E02CD0000         mvi a,2 ! call ?bnksl      ; select extra bank
003C F177               pop psw ! mov m,a          ; save the byte
003E 230B               inx h ! dcx b              ; bump pointer, drop count
0040 78B1               mov a,b ! ora c            ; test for done
0042 C23000             jnz ld$1
0045 F1CD0000           pop psw ! call ?bnksl      ; restore original bank
0049 C9                 ret

                no$CCP:                 ; here if we couldn't find the file
004A 21AB00CD00         lxi h,ccp$msg ! call ?pmsg ; report this...
0050 CD0000             call ?conin                ; get a response
0053 C30000             jmp ?ldccp                 ; and try again


                ?rlccp:
0056 2100010100         lxi h,0100h ! lxi b,0C00h  ; clone 3K
                rl$1:
005C 3E02CD0000         mvi a,2 ! call ?bnksl      ; select extra bank
0061 7EF5               mov a,m ! push psw         ; get a byte
0063 3E01CD0000         mvi a,tpa$bank ! call ?bnksl ; select TPA
0068 F177               pop psw ! mov m,a          ; save the byte
006A 230B               inx h ! dcx b              ; bump pointer, drop count
006C 78B1               mov a,b ! ora c            ; test for done
006E C25C00             jnz rl$1
0071 C9                 ret

                    ; No external clock.
                ?time:
0072 C9                 ret

                        ; CP/M BDOS Function Interfaces

                open:
0073 0E0FC30500         mvi c,15 ! jmp bdos        ; open file control block

                setdma:
0078 0E1AC30500         mvi c,26 ! jmp bdos        ; set data transfer address

                setmulti:
007D 0E2CC30500         mvi c,44 ! jmp bdos        ; set record count

                read:
0082 0E14C30500         mvi c,20 ! jmp bdos        ; read records


0087 0D0A0D0A43signon$msg       db      13,10,13,10,'CP/M Version 3.0, sample BIOS',13,10,0

00AB 0D0A42494Fccp$msg          db      13,10,'BIOS Err on A: No CCP.COM file',0
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
113
114  00CC 0143435020ccp$fcb      db    1,'CCP     ','COM',0,0,0,0
115  00DC                        ds    16
116  00EC 000000    fcb$nr       db    0,0,0
117
118  00EF 0326CFFF07init$table   db    3,p$zpio$3a,0CFh,0FFh,07h      ; set up config port
119  00F4 0327CF0007             db    3,p$zpio$3b,0CFh,000h,07h      ; set up bank port
120  00F9 012500                 db    1,p$bank$select,0      ; select bank 0
121  00FC 00                     db    0                      ; end of init$table
122
123  00FD                 end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

BANKED	FFFF	6#	18			
BC	0000					
BDOS	0005	16#	97	100	103	106
CCPFCB	00CC	50	52	56	114#	
CCPMSG	00AB	73	111#			
DE	0002					
FALSE	0000	4#				
FCBNR	00EC	51	116#			
HL	0004					
INITTABLE	00EF	30	118#			
IX	0004					
IY	0004					
LD1	0030	61#	68			
NOCCP	004A	53	72#			
OPEN	0073	52	96#			
OUTBLOCKS	0025	30	34#	39		
PBANKSELECT	0025	120				
PBAUDCON1	000C					
PBAUDCON2	0030					
PBAUDCON34	0031					
PBAUDLPT1	000E					
PBAUDLPT2	0032					
PBOOT	0014					
PCENTDATA	0011					
PCENTSTAT	0010					
PCON2DATA	002C					
PCON2STAT	002D					
PCON3DATA	002E					
PCON3STAT	002F					
PCON4DATA	002A					
PCON4STAT	002B					
PCONFIGURATION	0024					
PCRTDATA	001C					
PCRTSTAT	001D					
PFDCMND	0004					
PFDDATA	0007					
PFDINT	0008					
PFDMISC	0009					
PFDSECTOR	0006					
PFDSTAT	0004					
PFDTRACK	0005					
PINDEX	000F					
PLPT2DATA	0028					
PLPT2STAT	0029					
PLPTDATA	001E					
PLPTSTAT	001F					
PRTC	0033					
PSELECT	0008					
PWD1797	0004					
PZCTC1	000C					
PZCTC2	0030					
PZDART	001C					
PZDMA	0000					
PZPIO1	0008					
PZPIO1A	000A					
PZPIO1B	000B					

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

PZPIO2	0010						
PZPIO2A	0012						
PZPIO2B	0013						
PZPIO3	0024						
PZPIO3A	0026	118					
PZPIO3B	0027	119					
PZSIO1	0028						
PZSIO2	002C						
READ	0082	56	105#				
RL1	005C	80#	87				
SETDMA	0078	54	99#				
SETMULTI	007D	55	102#				
SIGNONMSG	0087	31	109#				
TPABANK	0001	19#	21#	62	83		
TRUE	FFFF	3#	4	6			
?BNKSL	0000	11	62	64	69	81	83
?CONIN	0000	9	74				
?INIT	0000	8	26#				
?LDCCP	0000	8	48#	75			
?PMSG	0000	9	31	73			
?RLCCP	0056	8	78#				
?TIME	0072	8	91#				
@AIVEC	0000	10	29				
@AOVEC	0000	10	29				
@CBNK	0000	11	60				
@CIVEC	0000	10	27				
@COVEC	0000	10	27				
@LOVEC	0000	10	28				

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

```
                        title 'Character I/O handler for z80 chip based system'

                ; Character I/O for the Modular CP/M 3 BIOS

                        ; limitations:

                        ;               baud rates 19200,7200,3600,1800 and 134
                        ;                       are approximations.

                        ;               9600 is the maximum baud rate that is likely
                        ;                       to work.

                        ;               baud rates 50, 75, and 110 are not supported


                        public  ?cinit,?ci,?co,?cist,?cost
                        public  @ctbl

                        maclib Z80      ; define Z80 op codes
                        maclib ports    ; define port addresses
                        maclib modebaud ; define mode bits and baud equates

0006 =          max$devices     equ 6

                        cseg

                ?cinit:
0000 79FE06CA42         mov a,c ! cpi max$devices ! jz cent$init ; init parallel printer
0006 D0                 rnc                             ; invalid device
0007 692600             mov l,c ! mvi h,0               ; make 16 bits from device number
000A E5                 push h                          ; save device in stack
000B 292929             dad h ! dad h ! dad h            ; *8
000E 11E900196E         lxi d,@ctbl+7 ! dad d ! mov l,m ; get baud rate
0013 7DFE07             mov a,l ! cpi baud$600           ; see if baud > 300
0016 3E44D21D00         mvi a,44h ! jnc hi$speed        ; if >= 600, use *16 mode
001B 3EC4               mvi a,0C4h                      ;    else, use *64 mode
                hi$speed:
001D 323501             sta sio$reg$4
0020 2600111B01         mvi h,0 ! lxi d,speed$table ! dad d     ; point to counter entry
0026 7E322E01           mov a,m ! sta speed             ; get and save ctc count
002A E1                 pop h                           ; recover
002B 11DC0019           lxi d,data$ports ! dad d        ; point at SIO port address
002F 7E3C323001         mov a,m ! inr a ! sta sio$port  ; get and save port
0034 11FAFF19           lxi d,baud$ports-data$ports ! dad d     ; offset to baud rate port
0038 7E322C01           mov a,m ! sta ctc$port          ; get and save
003C 212B01             lxi h,serial$init$tbl
003F C34500             jmp stream$out

                cent$init:
0042 213901             lxi h,pio$init$tbl

                stream$out:
0045 7EB7C8             mov a,m ! ora a ! rz
0048 47234E23           mov b,a ! inx h ! mov c,m ! inx h
                        outir
004C+EDB3               DB      0EDH,0B3H
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
 57  004E C34500          jmp stream$out
 58
 59
 60                ?ci:           ; character input
 61
 62  0051 78FE06D263       mov a,b ! cpi 6 ! jnc null$input ; can't read from centronics
 63                ci1:
 64  0057 CD6600CA57       call ?cist ! jz ci1             ; wait for character ready
 65  005D 0D               dcr c ! inp a                   ; get data
 66  005E+ED78             DB      0EDH,A*8+40H
 67  0060 E67F             ani 7Fh                         ; mask parity
 68  0062 C9               ret
 69
 70                null$input:
 71  0063 3E1A             mvi a,1Ah                       ; return a ctl-Z for no device
 72  0065 C9               ret
 73
 74                ?cist:         ; character input status
 75
 76  0066 78FE06D27D       mov a,b ! cpi 6 ! jnc null$status ; can't read from centronics
 77  006C 682600           mov l,b ! mvi h,0               ; make device number 16 bits
 78  006F 11DC0019         lxi d,data$ports ! dad d        ; make pointer to port address
 79  0073 4E0C             mov c,m ! inr c                 ; get SIO status port
 80                        inp a                           ; read from status port
 81  0075+ED78             DB      0EDH,A*8+40H
 82  0077 E601             ani 1                   ; isolate RxRdy
 83  0079 C8               rz                              ; return with zero
 84  007A F6FF             ori 0FFh
 85  007C C9               ret
 86
 87                null$status:
 88  007D AFC9             xra a ! ret
 89
 90                ?co:           ; character output
 91  007F 78FE06CA9E       mov a,b ! cpi 6 ! jz centronics$out
 92  0085 D29D00           jnc null$output
 93  0088 79F5             mov a,c ! push psw              ; save character from <C>
 94  008A C5               push b                          ; save device number
 95                co$spin:
 96  008B CDB300CA8B       call ?cost ! jz co$spin         ; wait for TxEmpty
 97  0091 E16C2600         pop h ! mov l,h ! mvi h,0       ; get device number in <HL>
 98  0095 11DC0019         lxi d,data$ports ! dad d        ; make address of port address
 99  0099 4E               mov c,m                         ; get port address
100  009A F1               pop psw ! outp a                ; send data
101  009B+ED79             DB      0EDH,A*8+41H
102                null$output:
103  009D C9               ret
104
105                centronics$out:
106  009E DB10E620C2       in p$centstat ! ani 20h ! jnz centronics$out
107  00A5 79D311           mov a,c ! out p$centdata        ; give printer data
108  00A8 DB10F601D3       in p$centstat ! ori 1 ! out p$centstat  ; set strobe
109  00AE E67ED310         ani 7Eh ! out p$centstat                ; clear strobe
110  00B2 C9               ret
111
112                ?cost:         ; character output status
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
00B3 78FE06CACD        mov a,b ! cpi 6 ! jz cent$stat
00B9 D27D00            jnc null$status
00BC 682600            mov l,b ! mvi h,0
00BF 11DC0019          lxi d,data$ports ! dad d
00C3 4E0C              mov c,m ! inr c
                       inp a                          ; get input status
00C5+ED78              DB      0EDH,A*8+40H
00C7 E604C8            ani 4 ! rz                     ; test transmitter empty
00CA F6FFC9            ori 0FFh ! ret                 ; return true if ready


                cent$stat:
00CD DB102F            in p$centstat ! cma
00D0 E620C8            ani 20h ! rz
00D3 F6FFC9            ori 0FFh ! ret

                baud$ports:             ; CTC ports by physical device number
00D6 0C0E3031          db      p$baud$con1,p$baud$lpt1,p$baud$con2,p$baud$con34
00DA 3132              db      p$baud$con34,p$baud$lpt2

                data$ports:             ; serial base ports by physical device number
00DC 1C1E2C2E          db      p$crt$data,p$lpt$data,p$con2data,p$con3data
00E0 2A28              db      p$con4data,p$lpt2data


00E2 4352542020@ctbl   db 'CRT   '     ; device 0, CRT port 0
00E8 0F                db mb$in$out+mb$serial+mb$softbaud
00E9 0E                db baud$9600
00EA 4C50542020        db 'LPT   '     ; device 1, LPT port 0
00F0 1F                db mb$in$out+mb$serial+mb$softbaud+mb$xon+xoff
00F1 0E                db baud$9600
00F2 4352543120        db 'CRT1  '     ; device 2, CRT port 1
00F8 0F                db mb$in$out+mb$serial+mb$softbaud
00F9 0E                db baud$9600
00FA 4352543220        db 'CRT2  '     ; device 3, CRT port 2
0100 0F                db mb$in$out+mb$serial+mb$softbaud
0101 0E                db baud$9600
0102 4352543320        db 'CRT3  '     ; device 4, CRT port 3
0108 0F                db mb$in$out+mb$serial+mb$softbaud
0109 0E                db baud$9600
010A 5641582020        db 'VAX   '     ; device 5, LPT port 1 used for VAX interface
0110 0F                db mb$in$out+mb$serial+mb$softbaud
0111 0E                db baud$9600
0112 43454E2020        db 'CEN   '     ; device 6, Centronics parallel printer
0118 02                db mb$output
0119 00                db baud$none
011A 00                db 0                    ; table terminator


011B 00FFFFFFE9speed$table     db      0,255,255,255,233,208,104,208,104,69,52,35,26,17,13,7


                serial$init$tbl
012B 02                        db 2            ; two bytes to CTC
012C            ctc$port       ds 1            ; port address of CTC
012D 47                        db 47h          ; CTC mode byte
012E            speed          ds 1            ; baud multiplier
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
169   012F 07                        db 7            ; 7 bytes to SIO
170   0130           sio$port        ds 1            ; port address of SIO
171   0131 1803E104                  db 18h,3,0E1h,4
172   0135           sio$reg$4       ds 1
173   0136 05EA                      db 5,0EAh
174   0138 00                        db 0            ; terminator
175
176   0139 02130F07  pio$init$tbl    db      2,p$zpio$2b,0Fh,07h
177   013D 0312CFF807                db      3,p$zpio$2a,0CFh,0F8h,07h
178   0142 00                        db 0
179
180   0143                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

BAUD110	0003						
BAUD1200	0008						
BAUD134	0004						
BAUD150	0005						
BAUD1800	0009						
BAUD19200	000F						
BAUD2400	000A						
BAUD300	0006						
BAUD3600	000B						
BAUD4800	000C						
BAUD50	0001						
BAUD600	0007	34					
BAUD7200	000D						
BAUD75	0002						
BAUD9600	000E	140	143	146	149	152	155
BAUDNONE	0000	158					
BAUDPORTS	00D6	44	129#				
BC	0000						
CENTINIT	0042	28	49#				
CENTRONICSOUT	009E	91	105#	106			
CENTSTAT	00CD	113	124#				
CI1	0057	63#	64				
COSPIN	008B	95#	96				
CTCPORT	012C	45	166#				
DATAPORTS	00DC	42	44	78	98	116	133#
DE	0002						
HISPEED	001D	35	37#				
HL	0004						
IX	0004						
IY	0004						
MAXDEVICES	0006	23#	28				
MBINOUT	0003	139	142	145	148	151	154
MBINPUT	0001						
MBOUTPUT	0002	157					
MBSERIAL	0008	139	142	145	148	151	154
MBSOFTBAUD	0004	139	142	145	148	151	154
MBXONXOFF	0010	142					
NULLINPUT	0063	62	70#				
NULLOUTPUT	009D	92	102#				
NULLSTATUS	007D	76	87#	114			
PBANKSELECT	0025						
PBAUDCON1	000C	130					
PBAUDCON2	0030	130					
PBAUDCON34	0031	130	131				
PBAUDLPT1	000E	130					
PBAUDLPT2	0032	131					
PBOOT	0014						
PCENTDATA	0011	107					
PCENTSTAT	0010	106	108	108	109	125	
PCON2DATA	002C	134					
PCON2STAT	002D						
PCON3DATA	002E	134					
PCON3STAT	002F						
PCON4DATA	002A	135					
PCON4STAT	002B						
PCONFIGURATION	0024						

CP/M PLUS V.3.0
COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

PCRTDATA	001C	134		
PCRTSTAT	001D			
PFDCMND	0004			
PFDDATA	0007			
PFDINT	0008			
PFDMISC	0009			
PFDSECTOR	0006			
PFDSTAT	0004			
PFDTRACK	0005			
PINDEX	000F			
PIOINITTBL	0139	50	176#	
PLPT2DATA	0028	135		
PLPT2STAT	0029			
PLPTDATA	001E	134		
PLPTSTAT	001F			
PRTC	0033			
PSELECT	0008			
PWD1797	0004			
PZCTC1	000C			
PZCTC2	0030			
PZDART	001C			
PZDMA	0000			
PZPIO1	0008			
PZPIO1A	000A			
PZPIO1B	000B			
PZPIO2	0010			
PZPIO2A	0012	177		
PZPIO2B	0013	176		
PZPIO3	0024			
PZPIO3A	0026			
PZPIO3B	0027			
PZSIO1	0028			
PZSIO2	002C			
SERIALINITTBL	012B	46	164#	
SIOPORT	0130	43	170#	
SIOREG4	0135	38	172#	
SPEED	012E	40	168#	
SPEEDTABLE	011B	39	162#	
STREAMOUT	0045	47	52#	57
?CI	0051	16	60#	
?CINIT	0000	16	27#	
?CIST	0066	16	64	74#
?CO	007F	16	90#	
?COST	00B3	16	96	112#
@CTBL	00E2	17	33	138#

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
1                       public @dtbl
2                       extrn fdsd0,fdsd1
3
4                       cseg
5
6   0000 00000000  @dtbl   dw fdsd0,fdsd1
7   0004 0000000000        dw 0,0,0,0,0,0,0,0,0,0,0,0,0,0          ; drives C-P non-existant
8
9   0020                   end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
FDSD0           0000    2       6
FDSD1           0000    2       6
@DTBL           0000    1       6#
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                        title 'wd1797 w/ Z80 DMA Single density diskette handler'

                ;   CP/M-80 Version 3     --  Modular BIOS

                ;       Disk I/O Module for wd1797 based diskette systems

                        ;       Initial version 0.01,
                        ;               Single density floppy only,     - jrp, 4 Aug 82

                        dseg

                    ; Disk drive dispatching tables for linked BIOS

                        public  fdsd0,fdsd1

                    ; Variables containing parameters passed by BDOS

                        extrn   @adrv,@rdrv
                        extrn   @dma,@trk,@sect
                        extrn   @dbnk

                    ; System Control Block variables

                        extrn   @ermde          ; BDOS error mode

                    ; Utility routines in standard BIOS

                        extrn   ?wboot  ; warm boot vector
                        extrn   ?pmsg   ; print message @<HL> up to 00, saves <BC> & <DE>
                        extrn   ?pdec   ; print binary number in <A> from 0 to 99.
                        extrn   ?pderr  ; print BIOS disk error header
                        extrn   ?conin,?cono    ; con in and out
                        extrn   ?const          ; get console status


                    ; Port Address Equates

                        maclib ports

                    ; CP/M 3 Disk definition macros

                        maclib cpm3

                    ; Z80 macro library instruction definitions

                        maclib z80

                    ; common control characters

000D =          cr      equ 13
000A =          lf      equ 10
0007 =          bell    equ 7


                    ; Extended Disk Parameter Headers (XPDHs)

```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
 57  0000 E600              dw      fd$write
 58  0002 DC00              dw      fd$read
 59  0004 DB00              dw      fd$login
 60  0006 BE00              dw      fd$init0
 61  0008 0000              db      0,0             ; relative drive zero
 62                 fdsd0   dph     trans,dpbsd,16,31
 63  000A+A400              DW TRANS                ; TRANSLATE TABLE ADDRESS
 64  000C+0000000000        DB 0,0,0,0,0,0,0,0,0,0  ; BDOS SCRATCH AREA
 65  0015+00                DB 0                    ; MEDIA FLAG
 66  0016+0000              DW DPBSD                        ; DISK PARAMETER BLOCK
 67  0018+2300              DW ??0001                       ; CHECKSUM VECTOR
 68  001A+3300              DW ??0002                       ; ALLOCATION VECTOR
 69  001C+FEFFFEFFFE        DW OFFFEH,OFFFEH,OFFFEH ; DIRBCB, DTABCB, HASH ALLOC'D
 70  0022+00                DB 0                    ; HASH BANK
 71  0023+          ??0001  DS      16              ; CHECKSUM VECTOR
 72  0033+          ??0002  DS      31              ; ALLOCATION VECTOR
 73
 74  0052 E600              dw      fd$write
 75  0054 DC00              dw      fd$read
 76  0056 DB00              dw      fd$login
 77  0058 CD00              dw      fd$init1
 78  005A 0100              db      1,0             ; relative drive one
 79                 fdsd1   dph     trans,dpbsd,16,31
 80  005C+A400              DW TRANS                ; TRANSLATE TABLE ADDRESS
 81  005E+0000000000        DB 0,0,0,0,0,0,0,0,0,0  ; BDOS SCRATCH AREA
 82  0067+00                DB 0                    ; MEDIA FLAG
 83  0068+0000              DW DPBSD                        ; DISK PARAMETER BLOCK
 84  006A+7500              DW ??0003                       ; CHECKSUM VECTOR
 85  006C+8500              DW ??0004                       ; ALLOCATION VECTOR
 86  006E+FEFFFEFFFE        DW OFFFEH,OFFFEH,OFFFEH ; DIRBCB, DTABCB, HASH ALLOC'D
 87  0074+00                DB 0                    ; HASH BANK
 88  0075+          ??0003  DS      16              ; CHECKSUM VECTOR
 89  0085+          ??0004  DS      31              ; ALLOCATION VECTOR
 90
 91                         cseg    ; DPB must be resident
 92
 93                 dpbsd   dpb 128,26,77,1024,64,2
 94  0000+1A00              DW      ??0005          ; 128 BYTE RECORDS PER TRACK
 95  0002+0307              DB      ??0006,??0007   ; BLOCK SHIFT AND MASK
 96  0004+00                DB      ??0008          ; EXTENT MASK
 97  0005+F200              DW      ??0009          ; MAXIMUM BLOCK NUMBER
 98  0007+3F00              DW      ??0010          ; MAXIMUM DIRECTORY ENTRY NUMBER
 99  0009+C000              DB      ??0011,??0012   ; ALLOC VECTOR FOR DIRECTORY
100  000B+1000              DW      ??0013          ; CHECKSUM SIZE
101  000D+0200              DW      2               ; OFFSET FOR SYSTEM TRACKS
102  000F+0000              DB      ??0014,??0015   ; PHYSICAL SECTOR SIZE SHIFT
103
104                         dseg    ; rest is banked
105
106                 trans   skew 26,6,1
107  00A4+01                DB      ?NXTSEC+1
108  00A5+07                DB      ?NXTSEC+1
109  00A6+0D                DB      ?NXTSEC+1
110  00A7+13                DB      ?NXTSEC+1
111  00A8+19                DB      ?NXTSEC+1
112  00A9+05                DB      ?NXTSEC+1
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
00AA+0B                 DB      ?NXTSEC+1
00AB+11                 DB      ?NXTSEC+1
00AC+17                 DB      ?NXTSEC+1
00AD+03                 DB      ?NXTSEC+1
00AE+09                 DB      ?NXTSEC+1
00AF+0F                 DB      ?NXTSEC+1
00B0+15                 DB      ?NXTSEC+1
00B1+02                 DB      ?NXTSEC+1
00B2+08                 DB      ?NXTSEC+1
00B3+0E                 DB      ?NXTSEC+1
00B4+14                 DB      ?NXTSEC+1
00B5+1A                 DB      ?NXTSEC+1
00B6+06                 DB      ?NXTSEC+1
00B7+0C                 DB      ?NXTSEC+1
00B8+12                 DB      ?NXTSEC+1
00B9+18                 DB      ?NXTSEC+1
00BA+04                 DB      ?NXTSEC+1
00BB+0A                 DB      ?NXTSEC+1
00BC+10                 DB      ?NXTSEC+1
00BD+16                 DB      ?NXTSEC+1


                    ; Disk I/O routines for standardized BIOS interface

                ; Initialization entry point.

                ;               called for first time initialization.


                fd$init0:
00BE 21CE00             lxi h,init$table
                fd$init$next:
00C1 7EB7C8             mov a,m ! ora a ! rz
00C4 47234E23           mov b,a ! inx h ! mov c,m ! inx h
                        outir
00CB+EDB3               DB      0EDH,0B3H
00CA C3C100             jmp fd$init$next

                fd$init1:       ; all initialization done by drive 0
00CD C9                 ret

00CE 040A       init$table      db 4,p$zpio$1A
00D0 CFC217FF                   db      11001111b, 11000010b, 00010111b,11111111b
00D4 040B                       db 4,p$zpio$1B
00D6 CFD017FF                   db      11001111b, 11011101b, 00010111b,11111111b
00DA 00                         db 0


                fd$login:
                                ; This entry is called when a logical drive is about to
                                ; be logged into for the purpose of density determination.

                                ; It may adjust the parameters contained in the disk
                                ; parameter header pointed at by <DE>
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
00DB C9              ret     ; we have nothing to do in
                             ;       simple single density only environment.


               ; disk READ and WRITE entry points.

                             ; these entries are called with the following arguments:

                                     ; relative drive number in @rdrv (8 bits)
                                     ; absolute drive number in @adrv (8 bits)
                                     ; disk transfer address in @dma (16 bits)
                                     ; disk transfer bank    in @dbnk (8 bits)
                                     ; disk track address    in @trk (16 bits)
                                     ; disk sector address   in @sect (16 bits)
                                     ; pointer to XDPH in <DE>

                             ; they transfer the appropriate data, perform retries
                             ; if necessary, then return an error code in <A>

               fd$read:
00DC 211802          lxi h,read$msg          ; point at ' Read '
00DF 3E8B0601        mvi a,8Bh ! mvi b,01h   ; 1797 read + Z80DMA direction
00E3 C3ED00          jmp rw$common

               fd$write:
00E6 211F02          lxi h,write$msg         ; point at ' Write '
00E9 3EA80605        mvi a,0A8h ! mvi b,05h  ; 1797 write + Z80DMA direction
                   ;   jmp wr$common

               rw$common:                    ; seek to correct track (if necessary),
                                             ;       initialize DMA controller,
                                             ;       and issue 1797 command.

00ED 222702          shld operation$name             ; save message for errors
00F0 321102          sta disk$command                ; save 1797 command
00F3 7832A802        mov a,b ! sta zdma$direction    ; save Z80DMA direction code
00F7 2A0000229F      lhld @dma ! shld  zdma$dma      ; get and save DMA address
00FD 3A00006F26      lda @rdrv ! mov l,a ! mvi h,0   ; get controller-relative disk drive
0103 11160219        lxi d,select$table ! dad d      ; point to select mask for drive
0107 7E321202        mov a,m ! sta select$mask       ; get select mask and save it
010B D308            out p$select                    ; select drive
               more$retries:
010D 0E0A            mvi c,10                        ; allow 10 retries
               retry$operation:
010F C5              push b                          ; save retry counter

0110 3A12022113      lda select$mask ! lxi h,old$select ! cmp m
0117 77              mov m,a
0118 C22D01          jnz new$track                   ; if not same drive as last, seek

011B 3A00002114      lda @trk ! lxi h,old$track ! cmp m
0122 77              mov m,a
0123 C22D01          jnz new$track                   ; if not same track, then seek

0126 DB09E602C2      in p$fdmisc ! ani 2 ! jnz same$track    ; head still loaded, we are OK
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                        new$track:      ; or drive or unloaded head means we should . . .
012D CDA901                     call check$seek         ; . . read address and seek if wrong track

0130 011B41                     lxi b,16667             ; 100 ms / (24 t states*250 ns)
                        spin$loop:                      ; wait for head/seek settling
0133 0B                         dcx b
0134 78B1                       mov a,b ! ora c
0136 C23301                     jnz spin$loop

                        same$track:
0139 3A0000D305                 lda @trk ! out p$fdtrack        ; give 1797 track
013E 3A0000D306                 lda @sect ! out p$fdsector      ;       and sector

0143 219A02                     lxi h,dma$block                 ; point to dma command block
0146 010011                     lxi b,dmab$length*256 + p$zdma  ; command block length and port address
                                outir                           ; send commands to Z80 DMA
0149+EDB3                       DB      0EDH,0B3H

014B DB25                       in p$bankselect                 ; get old value of bank select port
014D E63F47                     ani 3Fh ! mov b,a               ; mask off DMA bank and save
0150 3A00000F0F                 lda @dbnk ! rrc ! rrc           ; get DMA bank to 2 hi-order bits
0155 E6C0B0                     ani 0C0h ! ora b                ; merge with other bank stuff
0158 D325                       out p$bankselect                ; and select the correct DMA bank

015A 3A1102                     lda disk$command        ; get 1797 command
015D CDD501                     call exec$command       ; start it then wait for IREQ and read status
0160 321502                     sta disk$status         ; save status for error messages

0163 C1                         pop b                   ; recover retry counter
0164 B7C8                       ora a ! rz              ; check status and return to BDOS if no error

0166 E610                       ani 0001$0000b          ; see if record not found error
0168 C4A901                     cnz check$seek          ; if a record not found, we might need to seek

016B 0DC20F01                   dcr c ! jnz retry$operation

                        ; suppress error message if BDOS is returning errors to application...

016F 3A0000FEFF                 lda @ermde ! cpi 0FFh ! jz hard$error

                        ; Had permanent error, print message like:

                                ; BIOS Err on d: T-nn, S-mm, <operation> <type>, Retry ?

0177 CD0000                     call ?pderr             ; print message header

017A 2A2702CD00                 lhld operation$name ! call ?pmsg        ; last function

                                ; then, messages for all indicated error bits

0180 3A1502                     lda disk$status         ; get status byte from last error
0183 212902                     lxi h,error$table       ; point at table of message addresses
                        errml:
0186 5E235623                   mov e,m ! inx h ! mov d,m ! inx h ; get next message address
018A 87F5                       add a ! push psw        ; shift left and push residual bits with status
018C EBDC0000EB                 xchg ! cc ?pmsg ! xchg  ; print message, saving table pointer
```

COPYRIGHT (C) 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

```
0191 F1C28601           pop psw ! jnz errmi     ; if any more bits left, continue

0195 218A02CD00         lxi h,error$msg ! call ?pmsg    ; print '<BEL>, Retry (Y/N) ? '
019B CDF501             call u$conin$echo       ; get operator response
019E FE59CA0D01         cpi 'Y' ! jz more$retries ; Yes, then retry 10 more times
                hard$error:                     ; otherwise,
01A3 3E01C9             mvi a,1 ! ret           ;       return hard error to BDOS

                cancel:                         ; here to abort job
01A6 C30000             jmp ?wboot              ; leap directly to warmstart vector


                                ; subroutine to seek if on wrong track
                                ; called both to set up new track or drive

                check$seek:
01A9 C5                 push b                          ; save error counter
01AA CDE101             call read$id                    ; try to read ID, put track in <B>
01AD CABE01             jz id$ok                        ; if OK, we're OK
01B0 CDCE01             call step$out                   ; else step towards Trk 0
01B3 CDE101             call read$id                    ; and try again
01B6 CABE01             jz id$ok                        ; if OK, we're OK
01B9 CDD301             call restore                    ; else, restore the drive
01BC 0600               mvi b,0                         ; and make like we are at track 0
                id$ok:
01BE 78D305             mov a,b ! out p$fdtrack         ; send current track to track port
01C1 3A0000B8C1         lda @trk ! cmp b ! pop b ! rz   ; if its desired track, we are done
01C7 D307               out p$fddata                    ; else, desired track to data port
01C9 3E1A               mvi a,00011010b                 ; seek w/ 10 ms. steps
01CB C3D501             jmp exec$command


                step$out:
01CE 3E6A               mvi a,01101010b                 ; step out once at 10 ms.
01D0 C3D501             jmp exec$command

                restore:
01D3 3E0B               mvi a,00001011b                 ; restore at 15 ms
                        ; jmp exec$command


                exec$command:           ; issue 1797 command, and wait for IREQ
                                        ;       return status
01D5 D304               out p$fdcmnd                    ; send 1797 command
                wait$IREQ:                              ; spin til IREQ
01D7 DB08E640CA         in p$fdint ! ani 40h ! jz wait$IREQ
01DE DB04               in p$fdstat                     ; get 1797 status and clear IREQ
01E0 C9                 ret

                read$id:
01E1 21AB02             lxi h,read$id$block     ; set up DMA controller
01E4 01000F             lxi b,length$id$dmab$256 + p$zdma ; for READ ADDRESS operation
                        outir
01E7+EDB3               DB      0EDH,0B3H
01E9 3EC4               mvi a,11000100b         ; issue 1797 read address command
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
01EB CDD501          call exec$command      ; wait for IREQ and read status
01EE E69D            ani 10011101b          ; mask status
01F0 21110046        lxi h,id$buffer ! mov b,m      ; get actual track number in <B>
01F4 C9              ret                    ; and return with Z flag true for OK

                u$conin$echo:   ; get console input, echo it, and shift to upper case
01F5 CD0000B7CA      call ?const ! ora a ! jz u$c1   ; see if any char already struck
01FC CD0000C3F5      call ?conin ! jmp u$conin$echo  ; yes, eat it and try again
                u$c1:
0202 CD0000F5        call ?conin ! push psw
0206 4FCD0000        mov c,a ! call ?cono
020A F1FE61D8        pop psw ! cpi 'a' ! rc
020E D620            sui 'a'-'A'            ; make upper case
0210 C9              ret


0211          disk$command    ds      1       ; current wd1797 command
0212          select$mask     ds      1       ; current drive select code
0213          old$select      ds      1       ; last drive selected
0214          old$track       ds      1       ; last track seeked to

0215          disk$status     ds      1       ; last error status code for messages

0216 1020     select$table    db      0001$0000b,0010$0000b ; for now use drives C and D


                      ; error message components

0218 2C20526561read$msg       db      ', Read',0
021F 2C20577269write$msg      db      ', Write',0

0227 1802     operation$name  dw      read$msg

                      ; table of pointers to error message strings
                      ;       first entry is for bit 7 of 1797 status byte

0229 3902     error$table     dw      b7$msg
022B 4502                     dw      b6$msg
022D 4F02                     dw      b5$msg
022F 5702                     dw      b4$msg
0231 6A02                     dw      b3$msg
0233 7002                     dw      b2$msg
0235 7C02                     dw      b1$msg
0237 8302                     dw      b0$msg

0239 204E6F7420b7$msg         db      ' Not ready,',0
0245 2050726F74b6$msg         db      ' Protect,',0
024F 204661756Cb5$msg         db      ' Fault,',0
0257 205265636Fb4$msg         db      ' Record not found,',0
026A 204352432Cb3$msg         db      ' CRC,',0
0270 204C6F7374b2$msg         db      ' Lost data,',0
027C 2044524551b1$msg         db      ' DREQ,',0
0283 2042757379b0$msg         db      ' Busy,',0

028A 2052657472error$msg      db      ' Retry (Y/N) ? ',0
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
                        ; command string for Z80DMA device for normal operation
029A C3         dma$block       db      0C3h    ; reset DMA channel
029B 14                         db      14h     ; channel A is incrementing memory
029C 28                         db      28h     ; channel B is fixed port address
029D 8A                         db      8Ah     ; RDY is high, CE/ only, stop on EOB
029E 79                         db      79h     ; program all of ch. A, xfer B->A (temp)
029F            zdma$dma        ds      2       ; starting DMA address
02A1 7F00                       dw      128-1   ; 128 byte sectors in SD
02A3 85                         db      85h     ; xfer byte at a time, ch B is 8 bit address
02A4 07                         db      p$fddata ; ch B port address (1797 data port)
02A5 CF                         db      0CFh    ; load B as source register
02A6 05                         db      05h     ; xfer A->B
02A7 CF                         db      0CFh    ; load A as source register
02A8            zdma$direction  ds      1       ; either A->B or B->A
02A9 CF                         db      0CFh    ; load final source register
02AA 87                         db      87h     ; enable DMA channel
0011 =          dmab$length     equ     $-dma$block

02AB C3         read$id$block   db      0C3h    ; reset DMA channel
02AC 14                         db      14h     ; channel A is incrementing memory
02AD 28                         db      28h     ; channel B is fixed port address
02AE 8A                         db      8Ah     ; RDY is high, CE/ only, stop on EOB
02AF 7D                         db      7Dh     ; program all of ch. A, xfer A->B (temp)
02B0 1100                       dw      id$buffer ; starting DMA address
02B2 0500                       dw      6-1     ; Read ID always xfers 6 bytes
02B4 85                         db      85h     ; byte xfer, ch B is 8 bit address
02B5 07                         db      p$fddata ; ch B port address (1797 data port)
02B6 CF                         db      0CFh    ; load dest (currently source) register
02B7 01                         db      01h     ; xfer B->A
02B8 CF                         db      0CFh    ; load source register
02B9 87                         db      87h     ; enable DMA channel
000F =          length$id$dmab  equ     $-read$id$block

                        cseg    ; easier to put ID buffer in common

0011            id$buffer       ds      6       ; buffer to hold ID field
                        ; track
                        ; side
                        ; sector
                        ; length
                        ; CRC 1
                        ; CRC 2

0017                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

B0MSG 0283 381 390#
B1MSG 027C 380 389#
B2MSG 0270 379 388#
B3MSG 026A 378 387#
B4MSG 0257 377 386#
B5MSG 024F 376 385#
B6MSG 0245 375 384#
B7MSG 0239 374 383#
BC 0000
BELL 0007 52#
CANCEL 01A6 289#
CHECKSEEK 01A9 226 257 296#
CR 000D 50#
DE 0002
DISKCOMMAND 0211 203 249 354#
DISKSTATUS 0215 251 275 359#
DMABLENGTH 0011 239 412#
DMABLOCK 029A 238 398# 413
DPBSD 0000 62 66 79 83 93#
ERRM1 0186 277# 281
ERRORMSG 028A 283 392#
ERRORTABLE 0229 276 374#
EXECCOMMAND 01D5 250 310 316 323# 337
FDINIT0 00BE 60 143#
FDINIT1 00CD 77 152#
FDINITNEXT 00C1 145# 150
FDLOGIN 00DB 59 76 162#
FDREAD 00DC 58 75 188#
FDSD0 000A 14 62#
FDSD1 005C 14 79#
FDWRITE 00E6 57 74 193#
HARDERROR 01A3 263 286#
HL 0004
IDBUFFER 0011 339 422 434#
IDOK 01BE 299 302 305#
INITTABLE 00CE 144 155#
IX 0004
IY 0004
LENGTHIDDMAB 000F 333 430#
LF 000A 51#
MORERETRIES 010D 210# 285
NEWTRACK 012D 217 221 225#
OLDSELECT 0213 215 356#
OLDTRACK 0214 219 357#
OPERATIONNAME 0227 202 271 369#
PBANKSELECT 0025 243 247
PBAUDCON1 000C
PBAUDCON2 0030
PBAUDCON34 0031
PBAUDLPT1 000E
PBAUDLPT2 0032
PBOOT 0014
PCENTDATA 0011
PCENTSTAT 0010
PCON2DATA 002C
PCON2STAT 002D

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

PCON3DATA	00[illegible]					
PCON3STAT	002F					
PCON4DATA	002A					
PCON4STAT	002B					
PCONFIGURATION	0024					
PCRTDATA	001C					
PCRTSTAT	001D					
PFDCMND	0004	325				
PFDDATA	0007	308	406	425		
PFDINT	0008	327				
PFDMISC	0009	223				
PFDSECTOR	0006	236				
PFDSTAT	0004	328				
PFDTRACK	0005	235	306			
PINDEX	000F					
PLPT2DATA	0028					
PLPT2STAT	00[illegible]					
PLPTDATA	001E					
PLPTSTAT	001F					
PRTC	0033					
PSELECT	0008	209				
PWD1797	0004					
PZCTC1	000C					
PZCTC2	0030					
PZDART	001C					
PZDMA	0000	239	333			
PZPIO1	0008					
PZPIO1A	000A	155				
PZPIO1B	000B	157				
PZPIO2	0010					
PZPIO2A	0012					
PZPIO2B	0013					
PZPIO3	0024					
PZPIO3A	0026					
PZPIO3B	0027					
PZSIO1	0028					
PZSIO2	002C					
READID	01E1	298	301	331#		
READIDBLOCK	02AB	332	417#	430		
READMSG	0218	189	366#	369		
RESTORE	01D3	303	318#			
RETRYOPERATION	010F	212#	259			
RWCOMMON	00ED	191	198#			
SAMETRACK	0139	223	234#			
SELECTMASK	0212	208	215	355#		
SELECTTABLE	0216	207	361#			
SPINLOOP	0133	229#	232			
STEPOUT	01CE	300	314#			
TRANS	00A4	62	63	79	80	106#
UC1	0202	344	346#			
UCONINECHO	01F5	284	343#	345		
WAITIREQ	01D7	326#	327			
WRITEMSG	021F	194	367#			
ZDMADIRECTION	02A8	204	410#			
ZDMADMA	029F	205	403#			
?CONIN	0000	32	345	347		

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

?CONO	0000	32	348		
?CONST	0000	33	344		
?PDEC	0000	30			
?PDERR	0000	31	269		
?PMSG	0000	29	271	280	283
?WBOOT	0000	28	290		
@ADRV	0000	18			
@DBNK	0000	20	245		
@DMA	0000	19	205		
@ERMDE	0000	24	263		
@RDRV	0000	18	206		
@SECT	0000	19	236		
@TRK	0000	19	219	235	307

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                        title 'bank & move module for CP/M3 linked BIOS'

                        cseg

                        public ?move,?xmove,?bank
                        extrn @cbnk

                        maclib z80
                        maclib ports

                ?xmove: ; ALTOS can't perform interbank moves
0000 C9                 ret

                ?move:
0001 EB                 xchg            ; we are passed source in DE and dest in HL
                        ldir            ; use Z80 block move instruction
0002+EDB0               DB      0EDH,0B0H
0004 EB                 xchg            ; need next addresses in same regs
0005 C9                 ret

                                                        ; by exiting through bank select
                ?bank:
0006 C5                 push b                          ; save register b for temp
0007 171717E618         ral ! ral ! ral ! ani 18h       ; isolate bank in proper bit position
000C 47                 mov b,a                         ; save in reg B
000D DB25               in p$bankselect                 ; get old memory control byte
000F E6E7B0             ani 0E7h ! ora b                ; mask out old and merge in new
0012 D325               out p$bankselect                ; put new memory control byte
0014 C1                 pop b                           ; restore register b
0015 C9                 ret

                                                ;       128 bytes at a time

0016                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

BC	0000		
DE	0002		
HL	0004		
IX	0004		
IY	0004		
PBANKSELECT	0025	26	28
PBAUDCON1	000C		
PBAUDCON2	0030		
PBAUDCON34	0031		
PBAUDLPT1	000E		
PBAUDLPT2	0032		
PBOOT	0014		
PCENTDATA	0011		
PCENTSTAT	0010		
PCON2DATA	002C		
PCON2STAT	002D		
PCON3DATA	002E		
PCON3STAT	002F		
PCON4DATA	002A		
PCON4STAT	002B		
PCONFIGURATION	0024		
PCRTDATA	001C		
PCRTSTAT	001D		
PFDCMND	0004		

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

PFDDATA	0007
PFDINT	0008
PFDMISC	0009
PFDSECTOR	0006
PFDSTAT	0004
PFDTRACK	0005
PINDEX	000F
PLPT2DATA	0028
PLPT2STAT	0029
PLPTDATA	001E
PLPTSTAT	001F
PRTC	0033
PSELECT	0008
PWD1797	0004
PZCTC1	000C
PZCTC2	0030
PZDART	001C
PZDMA	0000
PZPIO1	0008
PZPIO1A	000A
PZPIO1B	000B
PZPIO2	0010
PZPIO2A	0012
PZPIO2B	0013

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

PZPIO3	0024		
PZPIO3A	0026		
PZPIO3B	0027		
PZSIO1	0028		
PZSIO2	002C		
?BANK	0006	5	22#
?MOVE	0001	5	14#
?XMOVE	0000	5	11#
@CBNK	0000	6	

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                        title 'System Control Block Definition for CP/M3 BIOS'

                        public @civec, @covec, @aivec, @aovec, @lovec, @bnkbf
                        public @crdma, @crdsk, @vinfo, @resel, @fx, @usrcd
                        public @mltio, @ermde, @erdsk, @media, @bflgs
                        public @date, @hour, @min, @sec, ?erjmp, @mxtpa


FE00 =          scb$base equ    0FE00H          ; Base of the SCB

FE22 =          @CIVEC  equ     scb$base+22h    ; Console Input Redirection
                                                ; Vector (word, r/w)
FE24 =          @COVEC  equ     scb$base+24h    ; Console Output Redirection
                                                ; Vector (word, r/w)
FE26 =          @AIVEC  equ     scb$base+26h    ; Auxiliary Input Redirection
                                                ; Vector (word, r/w)
FE28 =          @AOVEC  equ     scb$base+28h    ; Auxiliary Output Redirection
                                                ; Vector (word, r/w)
FE2A =          @LOVEC  equ     scb$base+2Ah    ; List Output Redirection
                                                ; Vector (word, r/w)
FE35 =          @BNKBF  equ     scb$base+35h    ; Address of 128 Byte Buffer
                                                ; for Banked BIOS (word, r/o)
FE3C =          @CRDMA  equ     scb$base+3Ch    ; Current DMA Address
                                                ; (word, r/o)
FE3E =          @CRDSK  equ     scb$base+3Eh    ; Current Disk (byte, r/o)
FE3F =          @VINFO  equ     scb$base+3Fh    ; BDOS Variable "INFO"
                                                ; (word, r/o)
FE41 =          @RESEL  equ     scb$base+41h    ; FCB Flag (byte, r/o)
FE43 =          @FX     equ     scb$base+43h    ; BDOS Function for Error
                                                ; Messages (byte, r/o)
FE44 =          @USRCD  equ     scb$base+44h    ; Current User Code (byte, r/o)
FE4A =          @MLTIO  equ     scb$base+4Ah    ; Current Multi-Sector Count
                                                ; (byte,r/w)
FE4B =          @ERMDE  equ     scb$base+4Bh    ; BDOS Error Mode (byte, r/o)
FE51 =          @ERDSK  equ     scb$base+51h    ; BDOS Error Disk (byte,r/o)
FE54 =          @MEDIA  equ     scb$base+54h    ; Set by BIOS to indicate
                                                ; open door (byte,r/w)
FE57 =          @BFLGS  equ     scb$base+57h    ; BDOS Message Size Flag (byte,r/o)
FE58 =          @DATE   equ     scb$base+58h    ; Date in Days Since 1 Jan 78
                                                ; (word, r/w)
FE5A =          @HOUR   equ     scb$base+5Ah    ; Hour in BCD (byte, r/w)
FE5B =          @MIN    equ     scb$base+5Bh    ; Minute in BCD (byte, r/w)
FE5C =          @SEC    equ     scb$base+5Ch    ; Second in BCD (byte, r/w)
FE5F =          ?ERJMP  equ     scb$base+5Fh    ; BDOS Error Message Jump
                                                ; (word, r/w)
FE62 =          @MXTPA  equ     scb$base+62h    ; Top of User TPA
                                                ; (address at 6,7)(word, r/o)
0000                    end
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

SCBBASE	FE00	9#	11	13	15	17	19	21	23	25	26
		28	29	31	32	34	35	36	38	39	41
		42	43	44	46						
?ERJMP	FE5F	6	44#								
@AIVEC	FE26	3	15#								
@AOVEC	FE28	3	17#								
@BFLGS	FE57	5	38#								
@BNKBF	FE35	3	21#								
@CIVEC	FE22	3	11#								
@COVEC	FE24	3	13#								
@CRDMA	FE3C	4	23#								
@CRDSK	FE3E	4	25#								
@DATE	FE58	6	39#								
@ERDSK	FE51	5	35#								
@ERMDE	FE4B	5	34#								
@FX	FE43	4	29#								
@HOUR	FE5A	6	41#								
@LOVEC	FE2A	3	19#								
@MEDIA	FE54	5	36#								
@MIN	FE5B	6	42#								
@MLTIO	FE4A	5	32#								
@MXTPA	FE62	6	46#								
@RESEL	FE41	4	28#								
@SEC	FE5C	6	43#								

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
@USRCD          FE44    4    31#
@VINFO          FE3F    4    26#
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
1                       ; equates for mode byte bit fields
2
3   0001 =      mb$input            equ 0000$0001b  ; device may do input
4   0002 =      mb$output           equ 0000$0010b  ; device may do output
5   0003 =      mb$in$out           equ mb$input+mb$output
6
7   0004 =      mb$soft$baud        equ 0000$0100b  ; software selectable
8                                                   ; baud rates
9
10  0008 =      mb$serial           equ 0000$1000b  ; device may use protocol
11  0010 =      mb$xon$xoff         equ 0001$0000b  ; XON/XOFF protocol
12                                                  ; enabled
13
14  0000 =      baud$none           equ 0           ; no baud rate associated
15                                                  ; with this device
16  0001 =      baud$50             equ 1           ; 50 baud
17  0002 =      baud$75             equ 2           ; 75 baud
18  0003 =      baud$110            equ 3           ; 110 baud
19  0004 =      baud$134            equ 4           ; 134.5 baud
20  0005 =      baud$150            equ 5           ; 150 baud
21  0006 =      baud$300            equ 6           ; 300 baud
22  0007 =      baud$600            equ 7           ; 600 baud
23  0008 =      baud$1200           equ 8           ; 1200 baud
24  0009 =      baud$1800           equ 9           ; 1800 baud
25  000A =      baud$2400           equ 10          ; 2400 baud
26  000B =      baud$3600           equ 11          ; 3600 baud
27  000C =      baud$4800           equ 12          ; 4800 baud
28  000D =      baud$7200           equ 13          ; 7200 baud
29  000E =      baud$9600           equ 14          ; 9600 baud
30  000F =      baud$19200          equ 15          ; 19.2K baud
31
32
```


CP/M - PLUS V3.0
COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

```
BAUD110      0003    18#
BAUD1200     0008    23#
BAUD134      000?    19#
BAUD150      0005    20#
BAUD1800     0009    24#
BAUD19200    000F    30#
BAUD2400     000A    25#
BAUD300      0006    21#
BAUD3600     000B    26#
BAUD4800     000C    27#
BAUD50       0001    16#
BAUD600      0007    22#
BAUD7200     000D    28#
BAUD75       0002    17#
BAUD9600     000E    29#
BAUDNONE     0000    14#
MBINOUT      0003     5#
MBINPUT      0001     3#    5
MBOUTPUT     0002     4#    5
MBSERIAL     0008    10#
MBSOFTBAUD   0004     7#
MBXONXOFF    0010    11#
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                ;    I/O Port addresses for Z80 chip set
                ;    based system with wd1797 FDC

                        ; chip bases

0000 =          p$zdma          equ 0
0004 =          p$wd1797        equ 4
0008 =          p$zpio1         equ 8
000C =          p$zctc1         equ 12
0010 =          p$zpio2         equ 16
0014 =          p$boot          equ 20  ; OUT disables boot EPROM
001C =          p$zdart         equ 28  ; console 1 and printer 1
0024 =          p$zpio3         equ 36
0028 =          p$zsio1         equ 40
002C =          p$zsio2         equ 44
0030 =          p$zctc2         equ 48


                        ; diskette controller chip ports

0004 =          p$fdcmnd        equ p$wd1797+0
0004 =          p$fdstat        equ p$wd1797+0
0005 =          p$fdtrack       equ p$wd1797+1
0006 =          p$fdsector      equ p$wd1797+2
0007 =          p$fddata        equ p$wd1797+3


                        ; parallel I/O 1

0008 =          p$select        equ p$zpio1+0
0008 =          p$fdint         equ p$zpio1+0
0009 =          p$fdmisc        equ p$zpio1+1
000A =          p$zpio1a        equ p$zpio1+2
000B =          p$zpio1b        equ p$zpio1+3


                        ; counter timer chip 1

000C =          p$baudcon1      equ p$zctc1+0
000E =          p$baud1pt1      equ p$zctc1+2
000F =          p$index         equ p$zctc1+3


                        ; parallel I/O 2, Centronics printer interface

0010 =          p$cent$stat     equ p$zpio2+0
0011 =          p$cent$data     equ p$zpio2+1
0012 =          p$zpio2a        equ p$zpio2+2
0013 =          p$zpio2b        equ p$zpio2+3


                        ; dual asynch rcvr/xmtr, console and serial printer ports

001C =          p$crt$data      equ p$zdart+0
001D =          p$crt$stat      equ p$zdart+1
001E =          p$lpt$data      equ p$zdart+2
001F =          p$lpt$stat      equ p$zdart+3
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
57
58                      ; Third Parallel I/O device
59
60  0024 =      p$configuration equ p$zpio3+0
61  0025 =      p$bankselect    equ p$zpio3+1
62  0026 =      p$zpio3a        equ p$zpio3+2
63  0027 =      p$zpio3b        equ p$zpio3+3
64
65
66                      ; Serial I/O device 1, printer 2 and console 4
67
68  0028 =      p$lpt2data      equ p$zsio1+0
69  0029 =      p$lpt2stat      equ p$zsio1+1
70  002A =      p$con4data      equ p$zsio1+2
71  002B =      p$con4stat      equ p$zsio1+3
72
73
74                      ; Serial I/O device 2, console 2 and 3
75
76  002C =      p$con2data      equ p$zsio2+0
77  002D =      p$con2stat      equ p$zsio2+1
78  002E =      p$con3data      equ p$zsio2+2
79  002F =      p$con3stat      equ p$zsio2+3
80
81
82                      ; second Counter Timer Circuit
83
84  0030 =     p$baudcon2       equ p$zctc2+0
85  0031 =     p$baudcon34      equ p$zctc2+1
86  0032 =     p$baudlpt2       equ p$zctc2+2
87  0033 =     p$rtc            equ p$zctc2+3
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

PBANKSELECT	0025	61#					
PBAUDCON1	000C	37#					
PBAUDCON2	0030	84#					
PBAUDCON34	0031	85#					
PBAUDLPT1	000E	38#					
PBAUDLPT2	0032	86#					
PBOOT	0014	11#					
PCENTDATA	0011	45#					
PCENTSTAT	0010	44#					
PCON2DATA	002C	76#					
PCON2STAT	002D	77#					
PCON3DATA	002E	78#					
PCON3STAT	002F	79#					
PCON4DATA	002A	70#					
PCON4STAT	002B	71#					
PCONFIGURATION	0024	60#					
PCRTDATA	001C	52#					
PCRTSTAT	001D	53#					
PFDCMND	0004	21#					
PFDDATA	0007	25#					
PFDINT	0008	30#					
PFDMISC	0009	31#					
PFDSECTOR	0006	24#					
PFDSTAT	0004	22#					
PFDTRACK	0005	23#					
PINDEX	000F	39#					
PLPT2DATA	0028	68#					
PLPT2STAT	0029	69#					
PLPTDATA	001E	54#					
PLPTSTAT	001F	55#					
PRTC	0033	87#					
PSELECT	0008	29#					
PWD1797	0004	7#	21	22	23	24	25
PZCTC1	000C	9#	37	38	39		
PZCTC2	0030	16#	84	85	86	87	
PZDART	001C	12#	52	53	54	55	
PZDMA	0000	6#					
PZPIO1	0008	8#	29	30	31	32	33
PZPIO1A	000A	32#					
PZPIO1B	000B	33#					
PZPIO2	0010	10#	44	45	46	47	
PZPIO2A	0012	46#					
PZPIO2B	0013	47#					
PZPIO3	0024	13#	60	61	62	63	
PZPIO3A	0026	62#					
PZPIO3B	0027	63#					
PZSIO1	0028	14#	68	69	70	71	
PZSIO2	002C	15#	76	77	78	79	

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____